Engineer

Leaflet

et lille bibliotek til kort i en browser

Hvad det er

Leaflet er et JavaScript-bibliotek med åben kildekode, der gør ét element på en webside til et kort, man kan panorere, zoome og klikke på. Projektet opgiver størrelsen til omkring 42 KB JavaScript og angiver ingen eksterne afhængigheder. Det blev skabt af Volodymyr Agafonkin, udvikles nu af et fællesskab af bidragydere og udgives under BSD 2-Clause-licensen.

Projektets egen oversigt siger, at det ikke forsøger at gøre alt for alle, men koncentrerer sig om at få det grundlæggende til at virke godt. `L.map` tager et element og et sæt indstillinger og bliver til kortet. Et flisekortlag henter kvadratiske billeder fra en URL-skabelon med `{z}`, `{x}` og `{y}` i og lægger dem ud som et gitter af billedelementer, og en WMS-variant gør det samme mod en Web Map Service. Polylinjer, polygoner, cirkler og rektangler tegnes som SVG som standard eller på ét enkelt canvas, når kortet oprettes med `preferCanvas`. Markører er ikoner, enten et billede eller en almindelig `div`, med popups og tooltips knyttet til. Et GeoJSON-lag gør hver feature til et af de lag, et punkt til en markør, medmindre andet angives. Zoomknapper, kildeangivelse, en lagvælger og en målestok udgør kontrollerne. Standardkoordinatsystemet er EPSG:3857, som referencen kalder det mest almindelige til onlinekort.

Alt ud over den liste kommer fra plugins. Projektets plugin-database beskriver Leaflet som tænkt til at være så letvægtigt som muligt og henviser til hundredvis af tredjeparts-plugins, sorteret i grupper som vektorfliser, klyngedannelse, heatmaps, geometriredigering, ruteplanlægning og geokodning.

To udgivelseslinjer findes i skrivende stund. Den stabile er 1.9.4, udgivet 18. maj 2023, og API-referencen på projektets site afspejler den. Forudgaven, 2.0.0-alpha.1 fra 16. august 2025, fjerner understøttelsen af Internet Explorer og

leveres som et ES-modul. Den beholder den globale `L` kun i en samlet `leaflet-global.js` , erstatter fabriksfunktioner med konstruktører, så `L.marker(latlng)` bliver til `new Marker(latlng)` , og tager input gennem pointer events.

Brug begrebet i en beslutning

Brug »Hvad er Leaflet?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Alt, der sker et sted — skibe, køretøjer, matrikler, sensorer — sat på et kort, I kan søge i, måle på og dele.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har gjort en geodatapipeline 50 gange hurtigere — 50x hurtigere geografiske datapipelines

Har gjort en geodatapipeline 50 gange hurtigere Kontakt os →

Hvorfor det betyder noget

At sætte et kort på en side er to beslutninger, der let forveksles med én: hvilken kode der tegner kortet i browseren, og hvor kortbillederne kommer fra. Leaflet holder dem adskilt, og at vælge Leaflet er for det meste et valg om at holde dem adskilt.

Leaflet tegner. Fliserne kommer derfra, hvor URL-skabelonen peger hen, så baggrundskortet er en særskilt beslutning med sin egen pris, sine egne vilkår og sin egen ejer. Det kan være OpenStreetMaps fliseserver, mens en funktion bygges, en kommerciel udbyder i produktion, en fliseserver, projektet selv kører, eller en kommunes WMS gennem `TileLayer.WMS` . At skifte mellem dem ændrer en streng og en linje med kildeangivelse, og applikationskoden omkring kortet bliver, som den er.

Størrelsen og licensen afgør, hvor Leaflet passer naturligt ind. Med omkring 42 KB og ingen afhængigheder er biblioteket velegnet til en side, hvor kortet er én del blandt mange: en formular, hvor nogen sætter en nål, eller en administrationsskærm, der viser, hvor posterne hører til. BSD 2-Clause-licensen sætter ingen konto, nøgle eller måling mellem biblioteket og siden.

Prisen for en lille kerne er, at tungere funktioner kommer som separate pakker, hver med sin egen vedligeholder og sit eget udgivelsestempo, og et kort, der hviler på seks plugins, arver seks opgraderingsveje. 2.0-alphaen viser, hvad det betyder: dens migreringsnoter henviser brugere af ældre plugins til en V1-polyfill, fordi de plugins forventer den globale `L` og fabriksfunktionerne. Et

kort, der har brug for vektorfliser stylet i browseren ved hvert zoomniveau, passer bedre i et bibliotek bygget omkring dem, og den beslutning træffes helst, før seks plugins holder kortet oppe.

Sådan ser det ud i praksis

Leaflet optræder på to sider på dette site, begge fra rollen som teamlead for GIS-softwareudvikling mellem 2015 og 2022, og på begge deler det frontenden med Mapbox.

Rollen bar leverancen på mere end tredive GIS-projekter, fra den første kode til supporten i drift. Foran i de projekter stod webkort bygget på Mapbox GL og Leaflet, nogle gange mod ArcGIS- eller HERE-API'erne, med app-laget i JavaScript, Python, PHP og SQL. Under dem lå PostgreSQL med PostGIS, og GDAL/OGR flyttede data mellem shapefiler, GeoJSON, GeoTIFF, KML og vektorfliser. Ansvaret rakte forbi leveringen og ind i supporten i produktion.

Den samme rolle sad også med i designprocessen for projekternes brugergrænseflader. Det var React-, Angular- og Vue-frontends oven på Mapbox- og Leaflet-kort, og siden gør en pointe, der er værd at holde ved siden af ethvert kortbibliotek: på et webkortprodukt er kortet den nemme del, og det er kontrollerne, filtreringen og flowet rundt om det, hvor folk farer vild. Brugervenligheden lå i detaljer som, hvordan et datasæt blev filtreret, og hvordan man skiftede mellem etager på et indendørskort, og siden tilskriver det meste af forskellen, at de spørgsmål blev stillet under design frem for efter release.

Hvor det går galt

Leaflet er så lille, at de fleste af dets fejl kommer fra siden omkring det eller fra de tjenester, det peger på.

Kortet vises som en grå kasse eller en stribe spredte fliser. Leaflet tager sin størrelse fra sin container. Quick start-guiden nævner to ting, der skal gøres, før nogen kortkode kører: indlæs Leaflets CSS-fil før JavaScriptet, og giv containeren en fastlagt højde. En `div` uden højde giver et kort uden højde. Et mere snigende tilfælde er et kort, der oprettes i en fane eller dialog, som er skjult på det tidspunkt: det måler sig selv til nul og beholder den størrelse, når det kommer frem. Leaflet 1.x følger med, når browservinduet ændrer størrelse, gennem `trackResize`, så en container, der ændrer størrelse af sig selv, går

ubemærket hen, indtil koden kalder `invalidateSize()`, som referencen siger skal kaldes, når kortets størrelse har ændret sig. 2.0-alphaen bruger `ResizeObserver` i stedet for vinduets resize-hændelse.

Punkter dukker op tusindvis af kilometer væk. GeoJSON, som RFC 7946 definerer det, skriver en position med længdegraden først. Leaflets `LatLng` og de arrays, biblioteket accepterer i stedet, har breddegraden først. `L.geoJSON` omregner selv mellem de to, så fejlen opstår, når koordinater løftes ud af GeoJSON i hånden og sendes direkte til en markør eller en polylinje. Et punkt i København med byttede koordinater lander i Det Arabiske Hav. Pak parret ud i navngivne variabler ved grænsen:

```
const res = await fetch("/api/sites/42");
const site = await res.json();
const [lon, lat] = site.geometry.coordinates; // GeoJSON: lon, lat
L.marker([lat, lon]).addTo(map); // Leaflet: lat, lng
```

Panorering hakker, når datamængden vokser. Med SVG-rendereren er hver linje eller polygon sit eget element i dokumentet, og hver standardmarkør er sit eget billede. Et lag med mange tusind features betyder mange tusind elementer, som browseren skal flytte ved hver panorering og zoom. `preferCanvas: true` tegner stier på ét canvas, men det gælder kun stier, så punkter skal laves om til `L.circleMarker`, en cirkel med fast radius i pixels, der selv er en sti, gennem GeoJSON-lagets `pointToLayer`-indstilling. Derfra er svaret at sende mindre: et klyngeplugin eller et API, der kun returnerer det, der passer til den aktuelle visning og zoom.

Fliserne holder op med at indlæse efter lancering. Quick start-guiden bygger sit første kort på OpenStreetMaps fliseserver og siger ligeud, at de fliser er fine til programmering, men at Tile Usage Policy skal læses før produktion. Politikken beder om synlig kildeangivelse, en User-Agent, der identificerer applikationen, og en gyldig Referer fra websider, caching efter HTTP-headerne og ingen massedownload eller forhåndshentning. Den giver ingen serviceaftale og kan blokere adgang uden varsel. Et travlt produktionskort hører hjemme hos en udbyder, hvis vilkår passer til dets trafik, eller på dets egen fliseserver, og med Leaflet består det skift af URL-skabelonen og kildeangivelsen.

Læs mere

- Leaflet — projektets oversigt, funktionsliste og et første kort på få linjer.

- Leaflet API reference — hver klasse, indstilling og metode i 1.9.4, herunder `preferCanvas`, `invalidateSize` og GeoJSON-indstillingerne.
- Leaflet Quick Start Guide — CSS'en, containerens højde og et første flisekortlag, trin for trin.
- Leaflet 2.0 Alpha released — hvad der ændrer sig i 2.0, og hvordan kode og plugins migreres.
- Tile Usage Policy — betingelserne for at bruge fliseserverne på openstreetmap.org.
- RFC 7946: The GeoJSON Format — GeoJSON-standarden, herunder at positioner skrives med længdegraden først.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er Leaflet?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/leaflet/